NOVEMBER/DECEMBER 1981
Volume ~~10~~ 9/Number 6

I.P. Sharp newsletter

Joint Venture with Singapore Airlines

On November 19, a new company started in Singapore. It is called Singapore International Software Services (SISS) and is 51% owned by Singapore Airlines (SIA) and 49% owned by I.P. Sharp Associates. The new company will sell SHARP APL time-sharing service and software in most of the Far Eastern countries, with the exception of Japan and Hong Kong. SIA will be installing SHARP APL on one of their computers during 1982 and customers in the Far East will have a choice of accessing the SIA machine or the main I.P. Sharp data centre in Toronto. The cost will be the same. SISS will have no staff of its own, but will draw on the resources of SIA and I.P. Sharp.

There are currently ten people in the SISS office. The general manager is **Hugh Hyndman**, formerly branch manager of the I.P. Sharp Singapore office. **Walter Keirstead**, who is in charge of I.P. Sharp's Far East headquarters, will assume the role of managing director of SISS.

This is the first time that I.P. Sharp Associates has embarked on a joint venture, and all indications so far are very positive. The signing ceremony was attended by about 150 people and received good coverage in the local press, radio and television.

Photo: At the signing ceremony; Cheong Choong Kong, SIA; Lim Chin Beng, SIA; Ian Sharp; Walter Keirstead.

CONTENTS

1 Joint Venture with Singapore Airlines

Data Bases
2 Comparing International Statistics using the BI and OECD Data Bases
7 New Stockmarket Data Base

Applications Software
5 SAGA

Meetings
8 APL Blossoms in SFO

Bulletin Board
10 Toronto, Boston, Atlanta, Seattle

Publications
11 Checklist, APL Press, APL 81 Proceedings

Technical Supplement 35
T1 Transpose
T3 Panel Discussion

COMPARING INTERNATIONAL STATISTICS USING THE BI AND OECD DATA BASES

Although there is no standard means of reporting international economic statistics, comparisons can now be made easily with the availability of two more international economic data bases—the Business International (BI) historical data base and the Organisation for Economic Co-operation and Development (OECD) data base.

Both data bases offer extensive economic statistics. The data is available to all users of the I.P. Sharp system with no surcharge or subscription fee. The BI data base is a collection of economic and demographic data for over 130 countries, with approximately 14,000 time series in all. The OECD data base contains domestic economic data describing the economies of its 25 member countries, with about 20,000 time series. When combined with the other economic data bases already available, a total of over 400,000 economic time series may be accessed by any user of the SHARP APL system.

BI data base

The BI historical data base contains annual time series for over 130 countries. The earliest data collected is for 1960. BI/DATA covers such categories as: demographic data, national income, gross domestic product, private consumption expenditure, capital formation, currency and balance of payments, labour force, production indices, components of manufacturing, wages, price trends, foreign trade and selected production and consumption data, including data for 43 specific industrial sectors. Within these categories, there are more than 180 accounts. For example, within the category *Hourly wages, manufacturing (local currency)* you can extract the average hourly wage for females, males or total population.

A prominent feature of BI/DATA is the power of the access function. At your discretion, the data may be reported by accounts within countries, or countries within accounts. To enable international statistics to be compared easily, any or all of three data conversions may be applied at the time of data retrieval. Currency data may be converted to any currency of the countries reported. Also many accounts may be expressed in terms of constant dollars (deflated to a base year). Thirdly, most accounts can be expressed in terms of the population (per capita). The access function with its syntax and options is fully documented in the on-line *describe* and is presented with a number of examples.

BI/DATA was the first on-line service of Business International Corporation. Business International was established in 1954 as an independent organization to provide business information to corporations doing business internationally as well as to academic institutions and government officials. BI offers executive, advisory, consulting, research and publishing services. Corporate headquarters is in New York, with the Latin American headquarters in Buenos Aires, the European in Geneva, and the Asian in Hong Kong. In addition, there are offices in 11 other major centres. Business International has a worldwide network of country and regional specialists, with resident editors, researchers or representatives in Bombay, Brussels, Jakarta, Rome and Stockholm and correspondents in 50 other cities.

Some of the sources of BI's data are the United Nations Monthly Bulletin of Statistics, OECD Economic Surveys, IMF's International Financial Statistics, Agency for International Development, Pick's Currency Yearbook, World Automotive Market, Johnston International Publishing Corporation, and the International Labour Organization Yearbook of Labour Statistics.

To achieve statistical consistency among countries between the latest published or estimated data and historical data, much of which have undergone substantial revision, BI analysts have relied on their own knowledge of each country to make estimates where necessary to bring the historical series up to date. The most current, broad-based statistical summary of international economic information is available from BI/DATA. Estimates are generally on the conservative side, and in preparing them, analysts have taken into account historical trends as well as current developments.

The following example illustrates the ease with which BI/DATA may be retrieved and displayed using MAGIC. In this example, the gross domestic product figures for six countries are accessed, using the *BIHIST* access function. The left argument to *BIHIST* specifies that the data is to be converted on a per capita basis to constant 1975 French francs. The right argument identifies the countries and account identifier for which data is required.

```
BISET
RESETOPTIONS
YEARLY, DATED 70 TO 80
AUTOLABEL
TITLE 'GROSS DOMESTIC PRODUCT - CONSTANT 1975 FRENCH FRANCS'
TABLE 'FRA/1975/PC' BIHIST 'FRA,GER,JAP,U.K,CAN,USB/GDPC'


      GROSS DOMESTIC PRODUCT - CONSTANT 1975 FRENCH FRANCS

                              PER-CAPITA
      ------------------------------------------------------
                                        UNITED              UNITED
          FRANCE  GERMANY    JAPAN      KINGDOM   CANADA    STATES
1970      23,566   26,770   16,170       16,063   26,081    28,517
1971      24,560   27,374   16,754       16,439   27,514    29,174
1972      25,780   28,198   18,089       16,894   28,781    30,560
1973      26,941   29,487   19,614       18,091   30,562    32,100
1974      27,620   29,554   19,250       17,866   31,147    31,697
1975      27,555   29,135   19,290       17,745   31,152    31,097
1976      28,855   30,698   20,323       18,485   32,532    32,535
1977      29,572   31,651   21,231       18,673   33,012    34,044
1978      30,566   32,928   22,313       19,366   33,804    35,389
1979      31,475   34,564   23,470       19,641   34,399    36,233
1980      31,876   35,229   24,606       19,170   33,517    35,871
```

OECD data base

The OECD data base describes the economies of the following countries: Australia, Austria, Belgium, Canada, Denmark, Finland, France, Germany, Greece, Iceland, Ireland, Italy, Japan, Luxembourg, Netherlands, New Zealand, Norway, Portugal, Spain, Sweden, Switzerland, Turkey, United Kingdom, United States, and Yugoslavia.

The data base consists of four sets of data: Main Economic Indicators (MEI), Quarterly National Accounts (QNA), Annual National Accounts—Volume 1 (ANA), and Annual National Accounts—Volume 2 (2 ANA). MEI data covers 29 countries and country groups in such areas as industrial production, deliveries, stocks and orders, construction, internal trade, labour and wages, and prices. This data base contains statistics collected on a monthly, quarterly and annual basis since 1960. QNA includes data from 12 countries for accounts such as gross domestic production, capital formation and private consumption. Data in QNA is available quarterly and annually since 1950.

ANA data from 29 countries and country groups includes time series describing government expenditures, private expenditures, exports of goods and services, indirect taxes, savings and national income. The data in Volume 1 of ANA dates back to 1950 on an annual basis while Volume 2 starts from 1960 on an annual basis. Volume 1 of ANA contains figures standardized by the OECD into a main aggregates table. In this way, international figures can be compared easily. Volume 2 contains data from 1960 in the form presented by the reporting country in more detailed tables.

Accessing the data

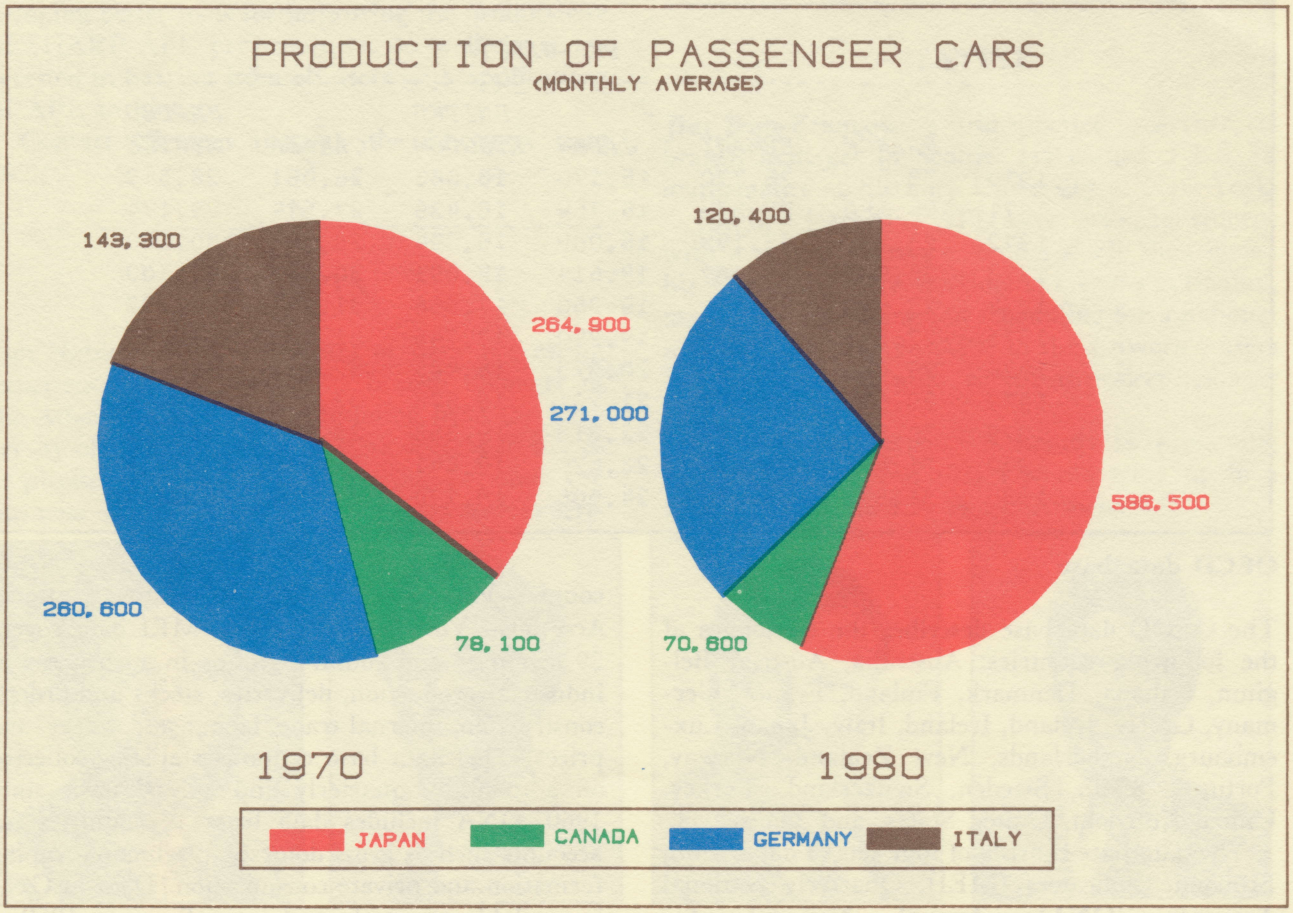
Both data bases are accessed through 39 *MAGIC*. The BI/DATA is available after typing *BISET*. An on-line directory and describe are available for high-speed or terminal printing, by typing *BIHIST 'DIRECTORY'* and *BIHIST 'DESCRIBE'*.

The OECD data is available through access functions *MEI* for the Main Economic Indicators, *QNA* for Quarterly National Accounts and *ANA* for Annual National Accounts. On-line directories are available by typing *MEI 'DIRECTORY'*, *QNA 'DIRECTORY'*, *ANA 'DIRECTORY'*, or 2 *ANA 'DIRECTORY'* for the respective data bases of the OECD. Also, an on-line describe is available by typing *MEI 'DESCRIBE'*, *QNA 'DESCRIBE'*, *ANA 'DESCRIBE'*, or 2 *ANA 'DESCRIBE'*, for the respective data bases. An added feature is the on-line cross-reference index which is accessed by typing *MEI 'XREF'* for the Main Economic Indicators data base and similarly, for the other data bases in this series.

The following example shows OECD data being used in conjunction with the new pie-charting capability of SUPERPLOT. In the next issue of the newsletter, we will expand on this new facility of SUPERPLOT to draw pie charts.

In this example, production of passenger cars for each of four countries is compared for the years 1970 and 1980 using the OECD Annual National Accounts data base.

Further information on both data bases is available through the MAILBOX group *ECOQ*.



SAGA

A Graphics Language for APL Programmers

Mike Holloway, Toronto

I.P. Sharp Associates has been involved in computer graphics for several years. As a result of the development of 3 *GRAPHICS* in the mid-seventies, and subsequent work on the SUPERPLOT system, a need was recognized for a low-level, device-independent graphics language. SAGA was designed to address this need. This tool for APL programmers developing graphics applications is becoming the foundation for the wide range of graphics packages currently being developed at I.P. Sharp.

What *is* computer graphics? Very simply, computer graphics consists of two parts: the **definition** (or creation) of pictures and their **display**. A **picture** is a general term referring to any type of drawing. The figures on the next page are all examples of pictures. How you define or create an arbitrary picture tends to vary depending on the type of picture. Therefore picture creation is best kept very specific to the application, whether the application is business graphics, cartography, or project planning.

Picture display, on the other hand, at its most basic level, consists of drawing **lines** with **text**, on a variety of **devices**—such as a plotter, terminal or CRT. All the various types of pictures you see here can really be viewed as lines with various characteristics—such as dotted or solid lines in colour—and text—in a variety of styles (or fonts).

Theoretically, a generalized computer application to handle the display of pictures—lines and text—is quite straightforward. In practice, the problem occurs when you want to be able to use the same commands to draw lines and text *on any device*.

Imagine if the method devised by car manufacturers to operate a car were different for *every* car manufacturer. In other words, what if the way you steer a car, accelerate and stop were not standard in all cars? Suppose not all cars used steering wheels; or in some cars, the position of the gas and brake pedals was reversed. At best, you would have more to learn before driving an unfamiliar car. At worst, the roads would be more dangerous than they already are. Fortunately, long ago, car manufacturers all over the world decided to standardize the way a driver operates a car.

Unfortunately, car manufacturers did not also design graphics devices. With graphics devices, there is *no* standard method of operating (or driving) the device, given even the most basic of instructions—such as "draw a straight black line". Thus it is not easy to design a basic set of graphics display commands capable of operating successfully *on any device*.

Having recognized this problem, I.P. Sharp Associates adopted a modular approach to the implementation of a basic set of graphics commands. To the user, there appears to be only one set of graphics commands, regardless of the device used.

But there's more to the basic set of commands than meets the eye. In fact, the underlying code for *each* command (such as "draw a straight black line") is different for *each* device supported by SAGA. A collection of these commands for a single device is called a **device driver**.

Thus, SAGA (**SHARP APL G**raphics **A**ids) is a graphics language designed to display pictures efficiently, while being independent of the device on which the picture is drawn. SAGA is becoming the cornerstone of all of I.P. Sharp's graphics development.

More about SAGA

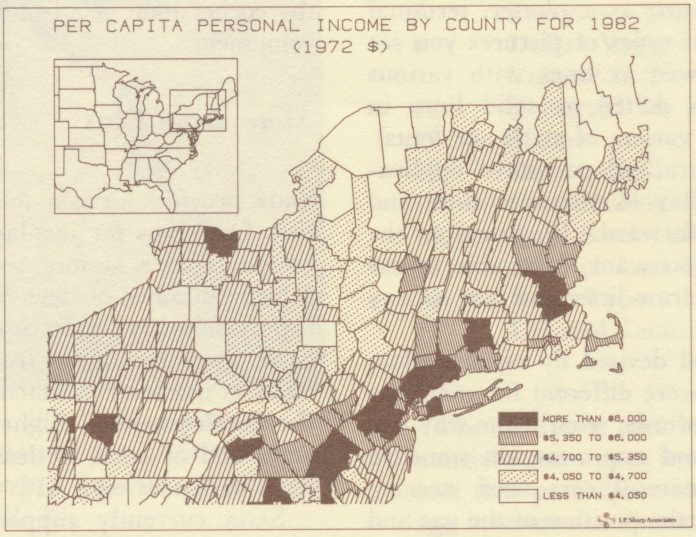
SAGA provides an easy-to-use set of **output primitive functions** for displaying arbitrary lines, rectangles, circles, sectors, segments, axes and text on a large number of devices. SAGA also provides an **input primitive function** that lets you perform basic graphics input from a variety of devices. SAGA is intended primarily as a tool to be used in the development of higher-level application packages, and as such, is designed to be used by systems designers and APL programmers.

SAGA currently supports 33 devices —flatbed plotters, hard-copy terminals and CRTs—with development underway for several more. By providing support for a large number of devices, SAGA relieves you of the responsibility of figuring out what characters must be transmitted to these devices to perform the basic graphics operations.

SAGA lets you alter the manner in which your pictures and text are displayed by providing a set of **primitive attribute functions**. Included in this set are functions to specify the pen colour, the line style, the size of text and many other attributes involved in picture display.

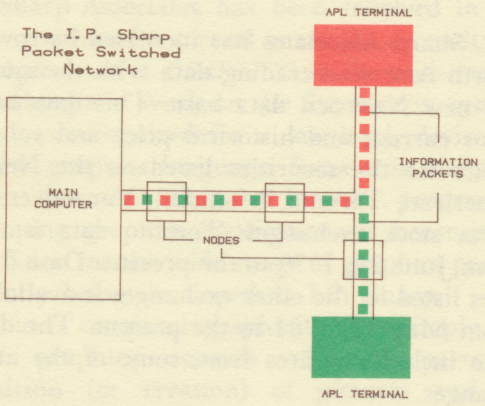
APL programmers designing graphics applications for general use will find SAGA very useful.

For example, SAGA is being used in the implementaton of I.P. Sharp's general-purpose graphics applications. SUPERPLOT is a general-purpose business graphics package built using SAGA. Figure 1 shows an example of a SUPERPLOT.



Another group in I.P. Sharp is currently developing a generalized mapping system using SAGA. For example, figure 2 is a map displaying data from the U.S. National Planning Association economic data base.

SAGA can also be used in *production* graphics applications, where a system produces large numbers of highly specialized drawings of a similar nature (not necessarily plots) on a regular basis. For example, SAGA could offer cost savings over a general-purpose system producing daily sales figure plots at the end of each month.



SAGA can also be used in ad hoc graphics applications. This might include preparing overhead transparencies for a presentation (see figure 3) or drawing unique designs such as your corporate logo.

Figure 4 shows a logo dear to all our hearts. In general, any application that may be considered "once-off" is ad hoc and can be dealt with efficiently using SAGA.

If you wish to know more about SAGA contact the MAILBOX group *SAGAQ* or your local I.P. Sharp representative.

SAGA is documented in the *SAGA Reference Manual* and the package resides in the workspace 3 *SAGA*.

APL BLOSSOMS IN SFO

Arlene Azzarello, Palo Alto

More than 800 registrants attended APL 81 at the San Francisco Hilton in October, and a straw vote indicates that it was the "best ever". Kudos to Gene Mannacio of Fireman's Fund and Dick Orgass of Xerox, and their dedicated committee members for providing a stimulating and amiable professional conference for the APL community. Speakers and attendees alike agreed that the local arrangements were superb and that APL 81 provided an excellent program.

At the top of the list of conference highlights were the quality of papers presented, the stimulating exhibits, and a first-rate banquet program. The development of an ANSI standard for APL created a great deal of interest. Both Clark Wiedmann's address reporting on its progress and the panel discussion with members of ANSI X3J10 were well-attended. Wiedmann, of STSC, is chairman of the committee. The number of exhibitors with full-blown APL systems on microcomputers was impressive. As well, Digital Effects, Inc. presented APL's role in computer animation for motion pictures and television, and in magazine graphics.

Underlining one of the salient points of the conference was STSC's eye-catching color hologram of a nested array. Three different approaches to nested or enclosed arrays emerged during this year's intensive immersion into APLdom. It was reminiscent of the simultaneous presentation, after years of debate, of three different APL file systems at the 1970 APL gathering at Goddard Space Center.

APL PRESS attracted a lot of attention with Pete McDonnell (definitely related to the publisher, Gene McDonnell) creating caricatures for an APL "Rogues' Gallery". Bob Bernecky of I.P. Sharp very aptly termed the burgeoning collection, the "APL Character Set", as his visage joined the likes of Ken Iverson of I.P. Sharp Associates and Adin Falkoff of IBM.

Roy Sykes of STSC deserved and got a big round of applause for a dynamite (you'll pardon the expression, but we *were* in California!) banquet program. Sykes had us rolling in the aisles, interested, and even participating in his APL games.

Everyone got to participate in one truly memorable event. Up from the APL Underground, the legendary J.C.L. Guest unleashed a new hit on the APL Top 40. Led by Larry Breed, Jim Brown, Diana Dloughy, Al O'Hara, and Rob Skinner, all from IBM, and John Bunda of Sperry-Univac, and with the assistance of Peter Wooster of I.P. Sharp and his overhead transparencies, we all found ourselves singing a new tune called "*APL* Blossom Time". The room echoed with laughter during this nostalgic reminiscence of the early days of APL sung to the tune of "The Battle of New Orleans". It's a tribute to J.C.L. Guest that it was hard for both APL veterans and newcomers alike to discern between being moved by nostalgia or by delight. Joy and wit do seem to go hand in hand. Hats off to J.C.L. Guest for entertaining us, once more!

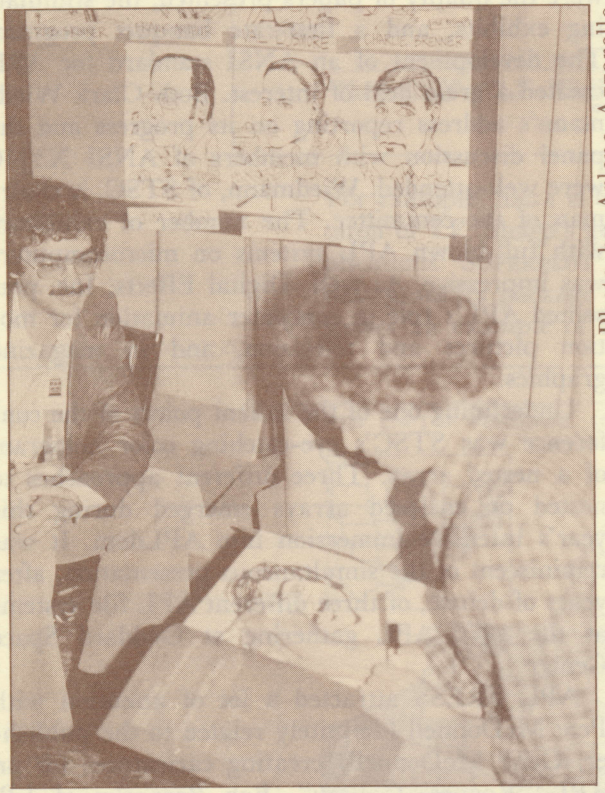


Photo by Arlene Azzarello

Leslie Goldsmith joins the "APL Character Set". Pete McDonnell (right) sketches Goldsmith for the APL "Rogues' Gallery" featured at the APL PRESS exhibit at APL 81.

APL 81 Conference Proceedings, $21.00 U.S. for ACM members and $28.00 U.S. for nonmembers. May be ordered **prepaid** from ACM Order Department, P.O. Box 64145, Baltimore, Maryland 21264.

NEW STOCKMARKET DATA BASE

Marc Odho, Toronto

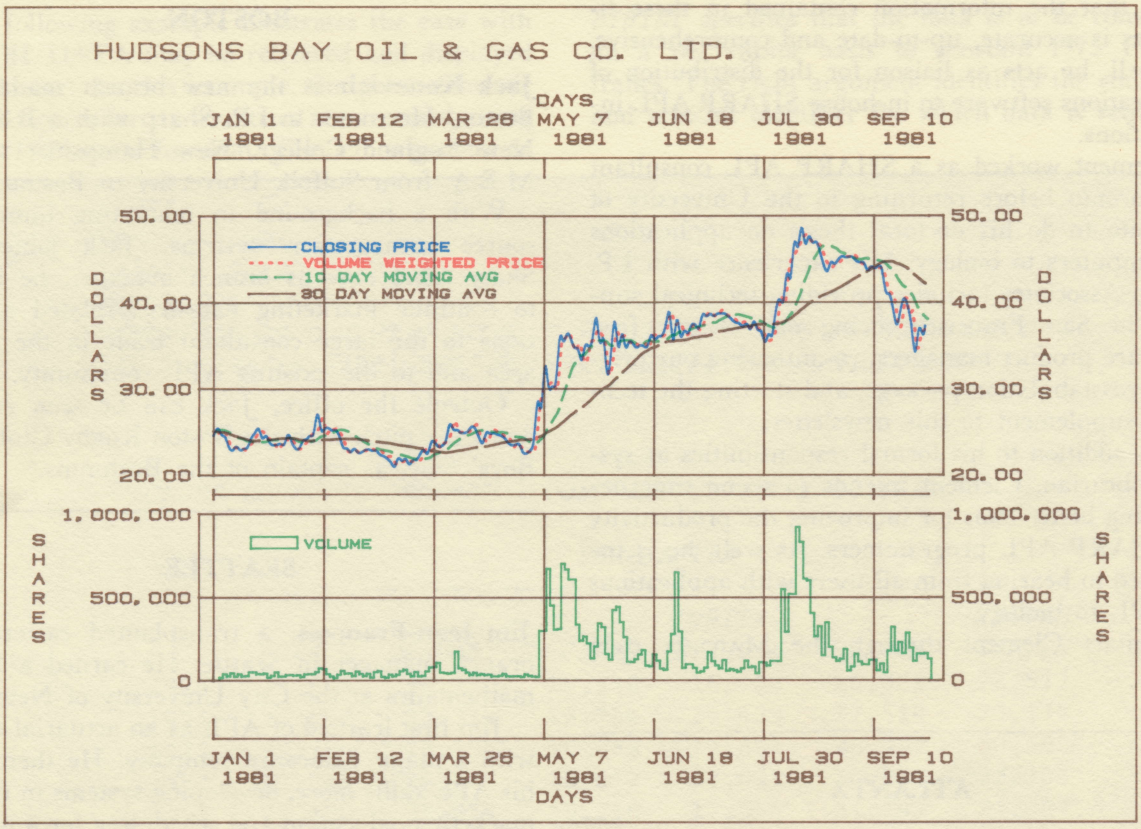
I.P. Sharp Associates has increased its coverage of North American trading data with the addition of the new NASTOCK data base. This data base contains current and historical price and volume statistics for the securities listed on the New York, American, Toronto, Montreal, Vancouver and Alberta stock exchanges. Toronto data is available from June 29, 1979 to the present. Data for securities listed on the other exchanges is available daily from May 22, 1981 to the present. The data base also includes indices from some of the above exchanges.

For each security listed, the following categories of data can be selected:

1	Bid Price
2	Ask Price
3	Open Price
4	High Price
5	Low Price
6	Close Price
7	Previous Close
8	Dollar Value
9	Volume
10	(Reserved for later use)
11	Symbol
12	Description
13	Classification (Industrials, Mines, Oils)
14	Date Stock Added
15	Date of Last Quote
16	Issue Type
17	Delisted Indicator

The NASTOCK data base will be of interest to financial institutions, investment houses, and companies that participate in the stock market, or manage investments. Particularly, it will be of interest to those involved in stock market research. This data base is suitable for many forms of analysis, including forecasting, moving average calculations, and standard deviation computations. The data can be used as an automatic input to portfolio systems and to produce output in tabular or graphic form.

The chart below was produced using data from the stockmarket data base in conjunction with I.P. Sharp's SUPERPLOT package. The chart illustrates the increase in the price of Hudson Bay Oil and Gas (HBO) shares, surrounding the acquisition of a 53% interest of HBO by Dome Petroleum from Conoco Inc. in June 1981. Notice the high volume during the months of April, May and June. The red line is the volume weighted closing price—that is, the total dollar value traded, divided by total volume. Shown also are 10 and 30 day moving averages. It is interesting to note that the *crossovers* between the moving average lines signal an increase or decrease in the closing price.

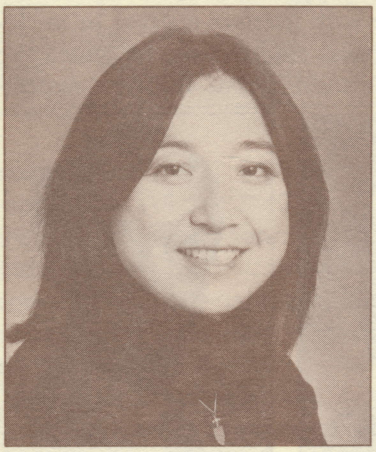


TORONTO

The *I.P. Sharp Newsletter* has a new editor. **Irene Shimoda** comes to the company with several years experience in corporate communications with a major data processing firm. There, Irene edited a marketing publication.

As editor of this newsletter, she hopes to stimulate and inform the SHARP APL community. She welcomes comments, articles, suggestions and barbs from all newsletter readers. As well, she hopes to maintain the high standard set by the previous editor, Jeanne Gershater.

Jeanne has given up editorial responsibilities to work in the System Development Group of I.P. Sharp Associates with Ken Iverson.



Contact Irene through the MAILBOX code *IRENE*.

TORONTO

Corporate headquarters in Toronto has a new system librarian. **Clement Kent** is responsible for managing, organizing and maintaining the quality control of all applications software in the I.P. Sharp public libraries. As system librarian for the SHARP APL time-sharing system, Clement ensures that the information contained in these libraries is accurate, up-to-date and comprehensive. As well, he acts as liaison for the distribution of applications software to in-house SHARP APL installations.

Clement worked as a SHARP APL consultant in Toronto before returning to the University of Toronto to do his doctoral thesis on applications of computers to biology. His experience with I.P. Sharp Associates includes providing technical support in San Francisco, being one of our first software product managers, co-authoring our original crosstabulation package, and starting the technical supplement to this newsletter.

In addition to his formal responsibilties as system librarian, Clement intends to spend time developing better tools for improving the productivity of SHARP APL programmers. As well, he is interested in hearing from all users with applications of APL to biology.

Contact Clement through the MAILBOX code *SYSL*.

ATLANTA

Jim Neblett brings 11 years of data processing experience to his position as branch manager of the Atlanta office. After graduating from the University of Louisville with masters degrees in nuclear physics and electrical engineering, he joined the business world. His most recent business experience includes two years as a stock broker and more than five years in marketing and management positions in the computer industry.

Jim has moved the Atlanta office to the city center, the better to serve existing clients and to accommodate new business in the southeast.

BOSTON

Jack Konevich is the new branch manager in Boston. He comes to I.P. Sharp with a B.S. from New England College, New Hampshire and an M.B.A. from Suffolk University in Boston.

With a background in marketing human resource management systems, Jack joined I.P. Sharp. And now as branch manager, he intends to continue marketing data-base-related applications to the large consultant trade in the Boston area and to the existing APL community.

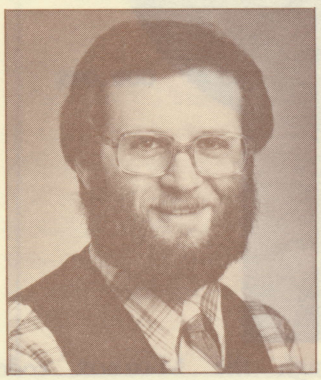
Outside the office, Jack can be seen running down the pitch with the Boston Rugby Club's Old Boys' Side, as captain of the Brahmins.

SEATTLE

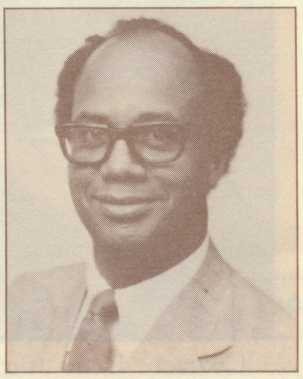
Jim Jean-Francois, a transplanted easterner, is branch manager in Seattle. He earned a B.S. in mathematics at the City University of New York.

Jim first learned of APL as an actuarial student with a major insurance company. He then honed his APL skills more, developing systems in finance, marketing, education and scheduling for a national network. He brought APL to the network's users on west coast, and after, served as project manager with GUIDE .

"Since my experience is mainly in promoting APL, it was natural that I would wind up in a company such as I.P. Sharp Associates."

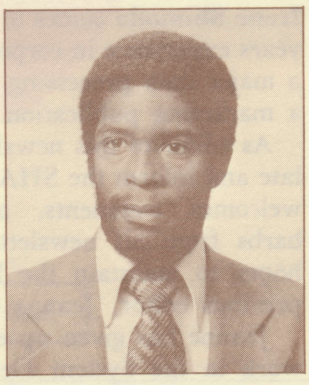


Clement Kent



Jim Neblett

Jack Konevich



Jim Jean-Francois

PUBLICATIONS CHECKLIST

Jane Minett, Toronto

It's time to update your SHARP APL bookshelf. If you would like copies of these publications, please contact your nearest I.P. Sharp office, or write to the Publications Department in Toronto.

SAGA Reference Manual (**S**HARP **A**PL **G**raphics **A**ids), *(available soon)*, October 81, 53 pp., $6. SAGA is a graphics display language for APL programmers. See the article on SAGA in this issue.

SNAP User's Manual(**S**harp **N**etwork **A**nalysis for **P**rojects), *(new)*, April 81, 259 pp., $15. SNAP is for project management. SNAP will help you plan, schedule, and keep track of the separate activities required in a project. See the article titled *Project Management* on page 3 of the *I.P. Sharp Newsletter*, September/October 1981.

SUPERPLOT User's Guide *(preliminary edition)* August 81, 119 pp. SUPERPLOT is a business graphics language for SHARP APL users without programming expertise. You can find examples of SUPERPLOT in the SAGA article and the *New Stock-market Data Base* article in this issue.

A NEW BOOK FROM APL PRESS

Arlene Azzarello, Palo Alto

Have you ever wondered why APL evaluates expressions from right to left? Or why **dieresis** is in the character set? Or why APL does without **declarations**? Answers to these and many more questions may be found in *A Source Book in APL*, a new book from APL PRESS. The purpose of the book is to provide background material for teachers and students of APL. In a course on APL, the focus is necessarily on the details of the language and its use. There is not always time to give the purpose for any given rule, nor how one piece of the language relates to the whole.

This new book is a collection of articles that deal with the more fundamental issues of the language. These papers appeared in widely scattered sources, over a period of many years, and are not always easy to find.

A Source Book in APL (139 pp., $8.95 U.S.) is available from APL PRESS, 220 California Avenue, Palo Alto, Ca. 94306 and the I.P. Sharp Publications Department in Toronto.

MAILING REQUEST

- ☐ Please amend my mailing address as indicated.
- ☐ Please add the following name(s) to your Newsletter mailing list.
- ☐ Please send me a Publications Order Form.
- ☐ Please add my name to the Energy Newsletter mailing list.
- ☐ Please add my name to the Aviation Newsletter mailing list.
- ☐ Please add my name to the Financial and Economic Newsletter mailing list.
- ☐ Please send me information about your courses in

(city) _____

Name: _____

Title: _____

Co.: _____

Address: _____

_____

_____

Telephone: _____

The Newsletter is a regular publication of I.P. Sharp Associates. Contributions and comments are welcome and should be addressed to: I.P. Sharp Newsletter, 145 King Street West, Toronto, Canada M5H 1J8.

Irene Shimoda, *Editor*
Mary Kopfensteiner, *Circulation*

Printed in Canada
December 1981
ISSN 0226 854X

I.P. Sharp Associates **Head Office:** 145 King Street West, Toronto, Canada M5H 1J8 (416) 364-5361

International Offices

Aberdeen
I.P. Sharp Associates Limited
5 Bon Accord Crescent
Aberdeen AB1 2DH
Scotland
(0224) 25298

Amsterdam
Intersystems B.V.
Kabelweg 47
1014 BA Amsterdam
The Netherlands
(020) 86 80 11
Telex: 18795 ITS NL

Atlanta
I.P. Sharp Associates, Inc.
1210 S. Omni International
Atlanta, Georgia 30335
(404) 586-9600

Boston
I.P. Sharp Associates, Inc.
Suite 415
148 State Street
Boston, Massachusetts 02109
(617) 523-2506

Bristol
I.P. Sharp Associates Limited
5 Whiteladies Rd., Clifton
Bristol BS8 1NN
(0272) 732 918

Brussels
I.P. Sharp Europe S.A.
Avenue du General de
Gaulle, 39
1050 Bruxelles
Belgique
(02) 649 99 77

Calgary
I.P. Sharp Associates Limited
Suite 550, Bow Valley Square 4
250-6th Ave. South West
Calgary, Alberta T2P 3H7 Canada
(403) 265-7730

Canberra
I.P. Sharp Associates Limited
16 National Circuit
Barton, ACT 2600
Australia
(062) 73-3700

Chicago
I.P. Sharp Associates, Inc.
2 North Riverside Plaza
Suite 1736
Chicago, Illinois 60606
(312) 648-1730

Cleveland
I.P. Sharp Associates, Inc.
(216) 431-6861
(local call, switched through
to Rochester office.)

Copenhagen
I.P. Sharp ApS
Ostergade 24B
1100 Copenhagen K
Denmark
(01) 11 24 34

Coventry
I.P. Sharp Associates Limited
7th Floor B Block
Coventry Point, Market Way
Coventry, England CV1 1EA
(0203) 56562

Dallas
I.P. Sharp Associates, Inc.
Suite 1148, Campbell Centre
8350 Northcentral Expressway
Dallas, Texas 75206
(214) 369-1131

Denver
I.P. Sharp Associates, Inc.
Suite 416
5680 South Syracuse Circle
Englewood, Colorado 80111
(303) 741-4404

Dublin
C/o Gamma Data Systems
Limited
Dollard House
Wellington Quay
Dublin 2, Ireland
(01) 731 877

Düsseldorf
I.P. Sharp GmbH
Leostrasse 62A
4000 Dusseldorf 11
West Germany
(0211) 57 50 16

Edmonton
I.P. Sharp Associates Limited
Suite 2358, Principal Plaza
10303 Jasper Avenue
Edmonton, Alberta T5J 3N6
(403) 428-6744

Gloucester
I.P. Sharp Associates Limited
29 Northgate Street
Gloucester, England GL1 2AN
(0452) 28106

Halifax
I.P. Sharp Associates Limited
Cogswell Tower
Suite 706, 2000 Barrington Street
Halifax, Nova Scotia
B3J 3K1, Canada
(902) 423-6251

Hamilton
I.P. Sharp Associates Limited
14 Hess South St., Hess Village
Hamilton, Ontario L8P 3M9
(416) 527-3801

Hong Kong
I.P. Sharp Associates (HK) Limited
Suite 606, Tower 1
Admiralty Centre, Hong Kong
5-294341

Houston
I.P. Sharp Associates, Inc.
Suite 375, One Corporate Square
2600 Southwest Freeway
Houston, Texas 77098
(713) 526-5275

Kitchener/Waterloo
I.P. Sharp Associates Limited
3 Menno St.
Waterloo, Ont. N2L 2A4
(519) 884-5420

London, Canada
I.P. Sharp Associates Limited
Suite 510, 220 Dundas Street
London, Ontario N6A 1H3
(519) 434-2426

London, England
(European Headquarters)
I.P. Sharp Associates Limited
132 Buckingham Palace Road
London SW1W 9SA, England
(01) 730-0361
Telex: 8954178 SHARP G

Los Angeles
I.P. Sharp Associates, Inc.
Suite 1230,
1801 Century Park East
Los Angeles, Ca. 90067
(213) 277-3878

Madrid
I.P. Sharp Associates Ltd.
Serrano 23, Piso 8
Madrid - 1, Spain
(91) 276 70 54

Manchester
I.P. Sharp Associates Limited
Paul House
89-91 Buttermarket Street
Warrington, Cheshire
England WA1 2NL
(0925) 50413/4

Melbourne
I.P. Sharp Associates Pty. Ltd.
520 Collins St., 13th Floor
Melbourne 3000
Victoria, Australia
(03) 614-1766

Mexico City
Teleinformatica de Mexico S.A.
(Agent) Mail to:
Arenal N 40, Chimalistac
Mexico 20 D.F., Mexico
(905) 550-8033

Miami
I.P. Sharp Associates, Inc.
Suite 240
15327 N.W. 60th Avenue
Miami Lakes, Florida 33014
(305) 556-0577

Milan
I.P. Sharp Srl (agent I.S.I.)
Via Eustachi 11
20129 Milan, Italy
(02) 221 612

Montreal
I.P. Sharp Associates Limited
Suite 1610
555 Dorchester Boulevard W.
Montreal, Quebec H2Z 1B1
(514) 866-4981

New York City
I.P. Sharp Associates Inc.
Suite 210
230 Park Avenue
New York, N.Y. 10169
(212) 557-1200

Newport Beach
I.P. Sharp Associates, Inc.
Suite 1135
610 Newport Center Drive
Newport Beach, Ca. 92660
(714) 644-5112

Oslo
I.P. Sharp A/S
Dronningens gate 34
Mail to: P. Boks 486 Sentrum
Oslo 1, Norway
(02) 41 17 04

Ottawa
I.P. Sharp Associates Limited
Suite 600, 265 Carling Ave.
Ottawa, Ontario K1S 2E1
(613) 236-9942

Palo Alto
I.P. Sharp Associates, Inc.
Suite 201, 220 California Ave.
Palo Alto, Ca. 94306
(415) 327-1700

Paris
I.P. Sharp Sarl
Tour Neptune, Cedex 20
20 Place de Seine
92086 Paris-la-defense
France
(1) 773 57 77

Philadelphia
I.P. Sharp Associates, Inc.
Suite 604, 437 Chestnut St.
Philadelphia, Pa. 19106
(215) 925-8010

Phoenix
I.P. Sharp Associates, Inc.
Suite 503
3033 N. Central Avenue
Phoenix, Arizona 85012
(602) 264-6819

Rochester
(United States Headquarters)
I.P. Sharp Associates, Inc.
(United States Headquarters)
1200 First Federal Plaza
Rochester, N.Y. 14614
(716) 546-7270
Telex: 0097 8473
0097 8474

San Francisco
I.P. Sharp Associates, Inc.
Suite C-415, 900 North Point St.
San Francisco, Ca. 94109
(415) 673-4930

San Jose
I.P. Sharp Associates, Inc.
3028A Scott Boulevard
Santa Clara, California 95050
(408) 727-9446

Saskatoon
I.P. Sharp Associates Limited
Suite 303, Financial Bldg.
230-22nd St. E. Saskatoon,
Saskatchewan S7K 0E9
(306) 664-4480

Seattle
I.P. Sharp Associates Inc.
Suite 223, Executive Plaza East
12835 Bellevue Redmond Road
Bellevue, Washington 98005,
(206) 453-1661

Singapore (Far East H.Q.)
I.P. Sharp Associates (S) Pte. Ltd.
PIL Building, 140 Cecil Street
Suite 10-03
Singapore 0106,
Republic of Singapore
2210494

Singapore
Singapore International
Software Services (SISS)
Suite 1501, CPF Building
79 Robinson Rd.
Singapore 0106.
Republic of Singapore
2230211

Stockholm
I.P. Sharp AB
Kungsgatan 65
S111 22 Stockholm, Sweden
(08) 21 10 19

Stuttgart/Boeblingen
I.P. Sharp GmbH
Schafgasse 3
7030 Boeblingen
West Germany
(070 31) 2 30 14

Sydney (Australian H.Q.)
I.P. Sharp Associates Pty. Ltd.
8th Floor Carlton Centre
55 Elizabeth St.
Sydney, New South Wales 2000,
Australia
(02) 232-6366

Toronto
(International Headquarters)
I.P. Sharp Associates Limited
145 King Street West
Toronto, Ontario M5H 1J8
(416) 364-5361

Toronto (Special Systems Div.)
I.P. Sharp Associates Limited
156 Front Street W., 5th Floor
Toronto, Ontario M5J 1G6
(416) 364-5361

Vancouver
I.P. Sharp Associates Limited
Suite 902, 700 West Pender St.
Vancouver, B.C. V6C 1G8
(604) 687-8991

Victoria
I.P. Sharp Associates Limited
Chancery Court
1218 Langley Street
Victoria, B.C. V8W 1W2
(604) 388-6365

Vienna
I.P. Sharp Ges.mbH
Rechte Wienzeile 5/3
A-1040 Wien, Austria
(0222) 57 65 71

Washington
I.P. Sharp Associates, Inc.
Suite 305, 2033 K Street N.W.
Washington, D.C. 20006
(202) 293-2915

White Plains
I.P. Sharp Associates Inc.
Suite 312 West, 701 Westchester Avenue
White Plains, New York 10604, U.S.A.
(914) 328-8520

Winnipeg
I.P. Sharp Associates Limited
Suite 208
213 Notre Dame Avenue
Winnipeg, Manitoba R3B 1N3
(204) 947-1241

Zurich
I.P. Sharp A.G.
Fortunagasse 15
8001 Zurich Switzerland
(01) 211 84 24

SHARP APL Communications Network

APL OPERATOR VOICE (416) 363-2051 COMMUNICATIONS (416) 363-1832

Local dial access is available in most of the locations listed above and in:

• Alliance • Ann Arbor • Austin • Baltimore • Birmingham • Buffalo • Clewiston (Fl) • Dayton • Des Moines • Des Plaines
• Detroit • Ft. Lauderdale • Geneva • Greenwich (Ct) • Hartford • Hull • Knoxville • Laurel • Leeds • Liverpool • Lyndhurst
• Minneapolis • Oxford • Quebec City • Raleigh • Red Deer • Regina • Santa Ana
• Sunnyvale • Syracuse • Towanda • Ukiah • Warrington

Our private, packet-switched network connects with the Value Added Networks in:

• Alaska • Argentina • Bahrain • Bermuda • Finland • Hawaii • Israel • Luxembourg • Mexico • New Zealand
• The Philippines • Portugal • Puerto Rico • Taiwan

In the continental United States, the SHARP APL Network is interconnected with the Value Added networks to provide access in 170 more cities, and in Canada with 40 more. In all, with the 80 cities served by the I.P. Sharp Network listed above, SHARP APL is accessible from close to 400 places via a local phone call. Please ask at your nearest I.P. Sharp office for a complete list of access points and access procedures. Our private network also connects with the worldwide Telex network via the Rochester. N.Y. and Amsterdam codes.

technical supplement 35

TRANSPOSE

Robert Metzger, Rochester

The transpose function (denoted by ⍉) is a useful, but often ignored, APL primitive. The purpose of this article is to explore the definition and application of monadic and dyadic transpose.

Most APL programmers seem to know how to use monadic transpose, at least on matrices. It reverses the axes of the argument array. For matrices, this means that rows become columns and columns become rows.

The most common use of monadic transpose is to take row-oriented data structures and set them up for column-oriented primitives. Encode and decode fall into the latter category. If you want to split a vector of numbers into their constituent digits, with one row representing each number, you can use the following function.

```
SPLIT: ⍉((1+⌊10⊛⌈/|⍵)⍴10)⊤|⍵
```

```
      SPLIT 1234 56 789
1 2 3 4
0 0 5 6
0 7 8 9
```

Here the encode puts the digits down the column of the result, and then transpose orients them in rows.

An inverse function must first orient the argument in columns. Then decode can return the original array.

```
PACK: 10⊥⍉⍵
```

```
      PACK SPLIT 1234 56 789
1234 56 789
```

Because encode and decode are column-oriented, but people tend to prefer to look at rows, you will frequently find transpose being used with encode and decode.

Inner product is the other primitive you often see used with monadic transpose. Of course, its definition says that it will apply the right function as a scalar function to a row of the left argument and a column of the right argument, and then apply the left function as reduction.

If the right argument is a row-oriented data structure, you will have to transpose it to get meaningful results with inner product. The function below shows one of the most common uses of inner product.

```
    ∇ Z←α LOOKUP ⍵;C
[1]   ⍵←(1⌈¯2↑⍴⍵)⍴⍵
[2]   C←(¯1↑⍴α)⌈¯1↑⍴⍵
[3]   Z←(((1↑⍴α),C)↑α)∧.=⍉((1↑⍴⍵),C)↑⍵
[4]   Z←(∨/Z)/⍳1↑⍴α ∇

      CAPITALS←5 6⍴'LONDONPARIS BONN  MADRIDROME '
      CAPITALS LOOKUP 2 5⍴'BONN PARIS'
2 3
```

While monadic transpose seems to get used quite a bit, dyadic transpose does not. The monadic case reverses the axes but dyadic transpose permutes the axes in a specified order. It can also select a cross-section as it is permuting the array.

The right argument is the array to be re-arranged. The left argument is an integer vector. Each of its elements must be one of the dimension numbers of the right argument, i.e., `∧/LEFT∈⍳⍴⍴RIGHT`. In addition, all axes of the result array must be specified in the left argument, i.e., `∧/(⍳⌈/LEFT)∈LEFT`. If all dimensions of the right argument are represented, i.e., `∧/(⍳⍴⍴RIGHT)∈LEFT`, then it will permute axes. If some dimensions of the right argument are not represented, i.e., `~∧/(⍳⍴⍴RIGHT)∈LEFT`, it will slice a cross-section.

Dimension shuffling

The number of possible dimension shuffles is `!⍴⍴RIGHT`. For a vector, you can only say `1⍉VECTOR`, which does nothing. For a matrix, you can say `1 2⍉MATRIX`, which does nothing, or `2 1⍉MATRIX`, which is the same as `⍉MATRIX`. This means that permuting axes with dyadic transpose is really only useful with arrays of rank 3 or greater. This may be part of the reason it isn't used as much as some other primitives.

What exactly is the definition of transpose when it just permutes axes? Well, the shape of the result will be determined by the rule that (ρ*RESULT*)∧.=(ρ*RIGHT*)[⍋*LEFT*]. This means that the values in the left argument refer to the axes of the result array, and the positions refer to the axes of the argument array. A number in the left argument is interpreted as follows: Move the axis specified by this position to the position specified by the number.

There are !3 = 6 ways to permute the axes of a cube. 1 2 3⍉*CUBE* just gives you the same cube back. It is always true that *CUBE*= _(⍳ρρ*CUBE*)⍉*CUBE*. 3 2 1⍉*CUBE* gives you ⍉*CUBE*. It is always true that (⍉*CUBE*)= _(⌽⍳ρρ*CUBE*)⍉*CUBE*. You can see that monadic transpose is simply a special case of dyadic transpose. If you want to explore this definition further, you can try the following.

```
EXAMPLE←(100×⍳3)∘.+(10×⍳4)∘.+⍳5
1 2 3⍉EXAMPLE ◇ 3 2 1⍉EXAMPLE
1 3 2⍉EXAMPLE ◇ 3 1 2⍉EXAMPLE
2 1 3⍉EXAMPLE ◇ 2 3 1⍉EXAMPLE
```

What can you use dyadic transpose for? One application is **collapsing dimensions**. This technique is used when we want to combine two or more axes of an array. Ravel collapses all the axes of an array into one, but sometimes we want to collapse only certain axes.

A function listed in a previous Technical Supplement (32) is an example of this technique.

```
    ∇ MAT←MATDOWN MAT;ROWS;COLS
[1]   MAT←' ',⍕MAT
[2]   COLS←⌊⎕PW÷¯1↑ρMAT
[3]   ROWS←⌈(1↑ρMAT)÷COLS
[4]   MAT← 2 1 3 ⍉(COLS,ROWS,¯1↑
      ρMAT)ρ((ROWS×COLS),¯1↑ρMAT)↑MAT
[5]   MAT←((1↑ρMAT),×/1↓ρMAT)ρMAT ∇
```

This program formats a matrix display so that multiple columns are displayed on a page. First it reshapes the matrix into a cube. Then it uses transpose to permute axes so that the appropriate data elements are adjacent. Finally, it collapses two dimensions with reshape to get the desired output.

A second application of transpose is **changing presentation order**. When you have a multidimensional array, the way APL presents that array on a two-dimensional surface may not be the best for you.

If you have a cube in which the first dimension is related to business locations, the second to products, and the third to months, there are six possible reports you could create with this method.

Page	Row	Column
Location	Product	Month
Location	Month	Product
Product	Location	Month
Product	Month	Location
Month	Location	Product
Month	Product	Location

The function listed below implements this idea. It allows you to prepare a report for a cube named *CUBE*. It assumes the existence of a character vector named *AXISLABELS*, which would have the value *'LPM'* (Location, Product, Month) in our example. It also expects that there will be a three-element vector named *AXISNAMES* in the workspace. It contains the enclosed values of the names of each subarray of each dimension. In the example given above,

```
AXISNAMES←(<LOCATIONNAMES),(<PRODUCTNAMES),(<MONTHNAMES).
```

It must be true that (ρ*CUBE*)∧.=(ρ⍤1>*AXISNAMES*)[;1].

If you want to run a report, you give the column width as the left argument and the order you want as the right argument. An example would be 10 *REPORT 'PLM'*.

```
    ∇ DIGITS REPORT ORDER
;DATA;NAMES;PAGEHEADINGS;WIDTH
;ROWHEADINGS;COLHEADINGS;CTR;LMT
[1]   ORDER←AXISLABELS⍳ORDER
[2]   DATA←(⍋ORDER)⍉CUBE
[3]   NAMES←AXISNAMES[ORDER]
[4]   PAGEHEADINGS←>NAMES[1]
[5]   ROWHEADINGS←>NAMES[2]
[6]   COLHEADINGS←>NAMES[3]
[7]   DIGITS←DIGITS⌈¯1↑ρCOLHEADINGS
[8]   COLHEADINGS←((¯1↑
      ρROWHEADINGS)ρ' '),,((1↑
      ρCOLHEADINGS),-DIGITS)↑
      RIGHTJUSTIFY COLHEADINGS
[9]   WIDTH←30⌈ρCOLHEADINGS
[10]  CTR←0
[11]  LMT←1↑ρDATA
[12] LOOP:→(LMT<CTR←CTR+1)ρEND
[13]  ⎕←CR,CR,(WIDTH CENTER
      PAGEHEADINGS[CTR;]),,CR,CR
      ,COLHEADINGS,CR,CR
[14]  ⎕←ROWHEADINGS,(DIGITS,0)
      ⍕DATA[CTR;;]
[15]  →LOOP
[16] END: ∇
```

You can see that the program must compute the appropriate left argument to transpose. While the axis names can be permuted directly with indexing, the order of the axes must be graded before applying transpose. This is necessary because `(ρRESULT)∧.=(ρRIGHT)[⍋LEFT]`. After the data has been put in the appropriate order, the rest of the function does the formatting and printing.

Dimension slicing

Transpose slices arrays by mapping two or more axes in the argument array into a single axis in the result array. More than one mapping can occur in a single application of transpose. The cross-section is selected by choosing those elements of the combined axes which have the same index position on each axis. Thus `1 1⍉MATRIX`, which selects the main diagonal, gets
`MATRIX[1;1],MATRIX[2;2],MATRIX[3;3],...`

For a cube, there are seven possible dimension slices you can obtain. Try these with the cube, *EXAMPLE*, assigned above.

```
1 1 1⍉EXAMPLE
1 1 2⍉EXAMPLE ◇ 2 2 1⍉EXAMPLE
1 2 1⍉EXAMPLE ◇ 2 1 2⍉EXAMPLE
1 2 2⍉EXAMPLE ◇ 2 1 1⍉EXAMPLE
```

There are 51 possible cross-sections of a four-dimensional array, and 421 possible cross-sections of a five-dimensional array. Can you come up with an APL expression to determine how many slices can be done on a rank-N array?

What can you do with dimension slicing? Unfortunately, most of the use it does get isn't well thought out. The most common usage is to implement a "middle product". By this people mean an operation which follows the shape conformability rules of inner product, but only applies a single function like outer product. An example would be multiplying a vector times each of the rows of a matrix. You might see this implemented as `1 2 2⍉MATRIX∘.×VECTOR`. This algorithm generates a tremendous amount of data which is just thrown away. A much less expensive algorithm is `MATRIX×(ρMATRIX)ρVECTOR`. Transpose cross-sections of outer product results can usually be done in another way which uses less time and space.

Acknowledgements: Special thanks to J. Henri Schueler for assistance in preparing this article.

PANEL DISCUSSION

on the Use of Enclosed Arrays

Roland Pesch, Palo Alto

One of the most eagerly awaited features of the APL 81 conference was a panel discussion with Ken Iverson of I.P. Sharp Associates, and Bob Smith of STSC Inc., on their respective companies' implementations of new APL features. These features, for both companies, are either related to enclosed arrays, or to new operators which are often useful in applications using enclosed arrays.

Don Orth of IBM introduced and moderated the discussion. He emphasized that what was taking place was a discussion, and not a debate (to the disappointment of many in the audience who had come hoping for fireworks). He proceeded to delineate clearly, in the context of several applications, the major differences between the STSC and I.P. Sharp extensions. Both Iverson and Smith also spoke to the major features of each implementation.

One major difference between the two is that of implementation strategy. STSC, according to Smith, has chosen to release a large number of extensions simultaneously on an "experimental research system" whose internal definitions are largely written in APL (although invoked by primitive symbols), rather than on their production service. They hope to learn from users' reactions what features are especially necessary or desirable. I.P. Sharp Associates, according to Iverson, has elected to move cautiously, implementing only the minimum necessary features, but integrating those that are implemented into its regular production release.

Technical differences

Many of the specific differences in the two implementations have to do with the definition of the function **enclose** when applied to a simple, non-enclosed scalar. In the I.P. Sharp system, the enclose ("strict enclose", denoted by <) of any object, including simple scalars, is a different object. In the STSC system, the enclose ("permissive enclose", denoted by ⊂) of a simple scalar is the same scalar. Smith, in particular, believes that many of the features of his system depended on that choice,

particularly "strand notation". Iverson pointed out that the I.P. Sharp system could allow the introduction of a permissive enclose (probably denoted by ≤) as well, but that it has been omitted for the nonce in order to defer the implementation complications of heterogeneous arrays.

Each of the two implementations also reflects different directions thought to be important in further developing APL. The STSC system involves the concepts of **type** and **prototype**; these are wholly absent from the I.P. Sharp implementation. **Prototypes** can be thought of as extensions to the difference (visible in the results of **take** and **expand**) between '' and ⍳0; they give rise to a large family of empty arrays.

The I.P. Sharp system involves concepts of **argument rank** and **application axes** for functions, which are wholly absent from the STSC implementation. These have little to do with enclosed arrays; rather, they specify the way in which functions (in particular those derived by the use of the new operators) apply to their arguments.

Differences summarized

Bob Bernecky, of I.P. Sharp, later summarized the differences between the two systems as follows: Smith sees nested arrays as the main datatype that APL applies to, with those we have known before now as merely special cases of it. Hence his implementation for STSC extends all APL primitives, whenever possible, to that domain. Iverson sees operators as the more general topic. Hence, the I.P. Sharp implementation counts on the use of operators to derive new functions as the vehicle for extending APL to the new domain of enclosed arrays.

Both speakers listed applications done to date with their new features. Both reported favorable reactions. Smith mentioned work with the STSC extensions was highly productive; having just completed a series of workshops, he remarked that it was easy to teach. Iverson reported users were favorably impressed by the simplicity of the concepts, and the simplicity of expressions using them. He reported, in the case of one application, done both with and without the new features, a reduction of 40% in the APL code involved. In terms of efficiency, he compared I.P. Sharp's enclosed arrays with packages, long a feature of the I.P. Sharp system, which have often been used in applications where enclosed arrays would also serve. He found that the space consumed by enclosed arrays was comparable to that consumed by packages, and that execution speed often improved with the use of enclosed arrays, particularly in expressions involving ⍳, ∈, or⌸.

Iverson also reported complaints about the current I.P. Sharp implementation, including lack of default display for enclosed arrays, and the incompleteness of the implementation to date. (A display form has now been implemented, and will probably be in the next release of SHARP APL.) Some users lamented the current lack of axis and partitioning operators, and the fact that operators don't yet accept derived or defined functions as arguments. All these extensions (described in the Bernecky and Iverson paper at the 1980 Users Meeting) are planned for future releases.

Smith mentioned that his system was intended to benefit from both negative and positive feedback, and that some improvements had already been made thanks to negative reactions. However, he did not list any specific complaints other than questions of performance.

Questions and answers

In the question-and-answer period, several members of the audience voiced concern about the fact that the two systems are so different. Both speakers stated that the availability of both was the best (perhaps the only) way for the language to evolve. Iverson, in particular, pointed to the history of mathematical notation, noting how things we now take for granted took a very long time to become generally accepted.

One questioner requested each speaker to comment on the consistency or desirability of the other's system. Both speakers emphasized that each system appeared to be internally consistent. Iverson added that as the I.P. Sharp version is much simpler and involves relatively few extensions, it is easy to see that there is no conflict with the rest of the language, whereas this might not be the case for a large set of extensions. Smith noted that all the additional features of his system were based on years of mathematical groundwork, and were therefore trustworthy.

Another question was whether each system's applications to date depended on the meaning of enclosing a simple scalar. Iverson and Bernecky said that to their knowledge, applications on the I.P. Sharp system didn't depend on this. Smith didn't allude to specific applications, but reiterated that a number of his extensions' definitions depended on this.